Gradient decsent for neural network.

Parameters: $W^{[1]}$, $b^{[1]}$, $W^{[2]}$, $b^{[2]}$

$(n^{[1]}, n^{[0]})$ $(n^{[1]}, 1)$ $(n^{[2]}, n^{[1]})$ $(n^{[2]}, 1)$

Cost function.: $J(W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}) = \frac{1}{m}\sum_{i=1}^{n} L(\hat{y}, y)$.

→ Gradient Descent:

Repeat:

Compute predicts: $(\hat{y}^{(i)}, \ldots, i-1, m)$.

$dW^{[1]} = \frac{\partial J}{\partial W^{[1]}}$, $db^{[1]} = \frac{\partial J}{\partial b^{[1]}}$, ... other partial derivatives.

$W^{[1]} := W^{[1]} - \alpha dW^{[1]}$

$b^{[1]} := b^{[1]} - \alpha db^{[1]}$

$W^{[2]} := W^{[2]} - \alpha dW^{[2]}$

... update all parameters using their partial derivatives and the learning rate.

Mathmetical Formulas:

forward propagation:

$Z^{[1]} = W^{[1]}X + b^{[1]}$

$A^{[1]} = g^{[1]}(Z^{[1]})$ These operations are all vectorized.

$Z^{[2]} = W^{[2]}x + b^{[2]}$

$A^{[2]} = g^{[2]}(Z^{[2]})$

$\equiv$

Back Propagation:

Rewind about Computing gradients for logistic Regression:

x, w, b $\rightarrow z = w^T x + b \rightarrow a = \sigma(z) \rightarrow \mathcal{L}(a, y)$

$dz = \frac{\partial \mathcal{L}}{\partial a} \cdot \frac{\partial a}{\partial z}$ $\quad da = \frac{d\mathcal{L}}{da} = -\frac{y}{a} + \frac{1-y}{1-a}$

$dw = dz \cdot x$

$db = dz$ $\quad = da \cdot g'(z) = a - y$

Back Propagation for Neural network:

x, $w^{[1]}$, $b^{[1]}$ $\rightarrow z^{[1]} = w^{[1]} x + b^{[1]} \rightarrow a^{[1]} = \sigma(z^{[1]})$

$da^{[1]} = \dots$

$dz^{[1]} = w^{[2]T} dz^{[2]}$

$dw^{[1]} = dz^{[1]} x^T$

$db^{[1]} = dz^{[1]}$

$w^{[2]}$, $b^{[2]}$ $\rightarrow z^{[2]} = w^{[2]} a^{[1]} + b^{[2]} \rightarrow a^{[2]} = \sigma(z^{[2]}) \rightarrow \mathcal{L}(a, y)$

$dz^{[2]} = a^{[2]} - y$

$dw^{[2]} = dz^{[2]} a^{[1]T}$

$db^{[2]} = dz^{[2]}$

$da^{[2]} = \frac{d\mathcal{L}}{da}$

Loss function is only used for prediction (last layer).

Summary of gradient descent.

$dz^{[2]} = a^{[2]} - y$

$dw^{[2]} = dz^{[2]} a^{[1]T}$

$db^{[2]} = dz^{[2]}$

$dz^{[1]} = w^{[2]T} dz^{[2]} * g^{[1]'}(z^{[1]})$

$dw^{[1]} = dz^{[1]} x^T$

$db^{[1]} = dz^{[1]}$

Vectorize Backward Propagation:

$$dZ^{[2]} = A^{[2]} - Y$$

$$dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]T}$$

$$db^{[2]} = \frac{1}{m} np.sum(dZ^{[2]}, axis=1, keepdims=True)$$

$$dZ^{[1]} = W^{[2]T} dZ^{[2]} * g^{[1]'}(Z^{[1]})$$

$$dW^{[1]} = \frac{1}{m} dZ^{[1]} X^{T}$$

$$db^{[1]} = \frac{1}{m} np.sum(dZ^{[1]}, axis=1, keepdims=True)$$